MULTIMEDIA UNIVERSITY

STUDENT ID NO

VENUE : _____

SEAT NO : _____

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 1, 2019/2020

TCP 1101 – PROGRAMMING FUNDAMENTALS

(All sections / Groups)

23 OCT 2019
9.00 a.m. – 11.00 a.m.
(2 Hours)

Question	Mark
1	
2	
3	
4	
Total	

INSTRUCTIONS TO STUDENT

1. This question paper consists of 9 printed pages (including cover page).

2. Attempt all questions. The distribution of the marks for each question is given.

3. Print all your answers **CLEARLY** in the specific answer box provided for each question. Submit this question paper at the end of the examination.

QUESTION 1 [10 marks]

(a) Why does the following statement give compilation error?

```
auto value;
```

[1]

(b) Assume a string object has been defined as follows.

```
string description;
```

i)Write a `cin` statement that reads in a one word string.
ii)Write a `getline` statement that reads in a string that can contain multiple words separated by blanks.

[2]

```
i)

ii)
```

(c) Complete the following table by providing another way of writing the same statements. The statements should be equivalent to the statements in the left hand column.

[3]

Statements	Equivalent Statements
`num += 2;`	
`people--;`	
`++people;`	

(d) You are asked by your introductory statistics lecturer to write a program that the lecturer can use in class to simulate the random generation of an integer value. The program should randomly generate one integer value in the range of 20 to 40 and display it. Complete the program in the box provided below. Use the rand() function that returns a pseudo-random integral number in the range between 0 and 32767. Include the statement srand(time(0)) in your program.

[4]

```
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;

int main()
{




}
```

Continued...

QUESTION 2 [10 marks]

(a) Complete the program below using a *for-loop* to keep track of products in an inventory system. Starting from product number 500, print out a line for every 10 products until all products have been tracked. Print an additional line when tracking is completed. The expected output is provided below.

Sample Run:

```
Product #500 tracked.
Product #490 tracked.
Product #480 tracked.
...     ← The program continues to display until Product #10
...
...
Product #20 tracked.
Product #10 tracked.
Tracking complete.
```

[3]

```
#include <iostream>
using namespace std;

int main()
{
   int items = 500;       // USE this variable









   return 0;
}
```

(b) Write the function `vowelswap` to swap two characters only if they consist of vowels 'a' and 'e'. The order of input is not important. Some expected outputs are provided below (user input in bold).

Sample Run 1 to 4

Enter two characters: **e a** a e	Enter two characters: **a e** e a
Enter two characters: **z e** z e	Enter two characters: **g q** g q

[4]

```
#include <iostream>
using namespace std;
```

Continued...

```
int main(){
    char c1, c2;
    cout << "Enter two characters: ";
    cin >> c1 >> c2;
    vowelswap(c1, c2);
    cout << c1 << " " << c2;
}
```

(c) Identify the syntax/logical errors in the following program which calculates 3^4. Write the line number where the error occurs with a brief description of the error.

```
1    #include <iostream>
2    #include <string>
3    using namespace std;
4
5    int main(){
6        double a = 3;
7        int n = 4;
8        p = power(a, n);
9        cout << p;
10   }
11
12   double power(double base, int exp)
13   {
14       double prod = 0;
15       while(exp > 0);
16       {
17           prod *= base;
18           exp = exp - 1;
19       }
20       return prod;
21   }
```

[3]

Line	Description of Error

Continued...

QUESTION 3 [10 marks]

(a) Complete the following C++ program.

Sample Run :

```
Total Area = 2.8
```

[4]

```
#include <iostream>
#include <vector>
using namespace std;

class Square
{
   __________

      //declare constructor: default dimension is 1.0

      ____________________________________

      double getArea();

   private:
      double dim;
};

Square::Square(double dimension)
{
   dim = dimension;
}

_____________________________________
{
   return dim * dim;
}

int main()
{
   Square s1(0.6), s2(1.2);
   Square s3;

   double totalArea = s1.getArea() + s2.getArea() + s3.getArea();
   cout << "Total Area = " << totalArea;
}
```

(b) Given the class `Square` in (a) above, a new class `Squares` and a new `main()` function as shown below, complete the program so that it will produce the same output as the program in (a) above.

[6]

```
class Squares
{
   public:
      Squares()
      { }

      void addSquare( const ____________________ sq )
      {

         ____________________________________

      }
```

Continued...

```
      int getTotalSquare()
      {

         return ____________________
      }

      double getTotalArea();

   private:
      vector<Square> squareList;
};

double Squares::getTotalArea()
{
   double totalArea = 0.0;

   for( int i = 0; ____________________________; ++i )
   {

      totalArea += ____________________________
   }
   return totalArea;
}

int main()
{
   Square s1(0.6), s2(1.2);
   Square s3;

   Squares squares;
   squares.addSquare(s1);
   squares.addSquare(s2);
   squares.addSquare(s3);
   cout << "Total Area = " << squares.getTotalArea();
}
```

Continued...

QUESTION 4 [10 marks]

(a) Write the code to implement `expandArray()` which doubles the size of a dynamic array. Assume that the user will always enter a valid array size when asked.

Sample Run 1:

```
Enter the size of the initial array: 3
The contents of the original array are:
1 2 3
The contents of the expanded array are:
1 2 3 0 0 0
```

Sample Run 2:

```
Enter the size of the initial array: 8
The contents of the original array are:
1 2 3 4 5 6 7 8
The contents of the expanded array are:
1 2 3 4 5 6 7 8 0 0 0 0 0 0 0 0
```

[5]

```
#include <iostream>
using namespace std;

int* expandArray( int* arr, int& size);
void showArray( int arr[], int size );

int main()
{
   int size;

   cout << "Enter the size of the initial array: ";
   cin >> size;

   int* data = new int[ size ];

   for( int i = 0; i < size; ++i )
   {
      data[ i ] = i + 1;
   }

   cout << "The contents of the original array are:" << endl;
   showArray( data, size );

   data = expandArray( data, size );

   cout << "The contents of the expanded array are:" << endl;
   showArray( data, size );

   delete [] data;
   data = nullptr;

   return 0;
}

void showArray( int arr[], int size )
{
   for( int i = 0; i < size; ++i )
   {
      cout << arr[ i ] << " ";
   }
   cout << endl;
}
```

Continued...

```
int* expandArray( int* arr, int& size )
{









}
```

(b) Complete the following program which writes ten integers into a binary file and then read the integers back from any position in the binary file. From the initial read position, the program reads forward the integers until the tenth integer in the file or reads backward the integers until the first integer in the file. Assume that the user will always enter valid input.

Sample Run 1:

```
Input playback startPosition ( 1 to 10 ) : 4
Input playback direction ( 1-Forward, 2-Backward ) : 1
4 5 6 7 8 9 10
```

Sample Run 2:

```
Input playback startPosition ( 1 to 10 ) : 5
Input playback direction ( 1-Forward, 2-Backward ) : 2
5 4 3 2 1
```

[5]

```
#include <iostream>
#include <fstream>
#include <cctype>
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
   const int SIZE = 10;
   int data[ SIZE ] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

   fstream file;

   file.open( "test.dat", ios::out | ios::binary);
   file.write( reinterpret_cast<char *>( data ), sizeof( data ) );
   file.close();
```

Continued...

```
    int playDirection;
    long int startPosition;
    cout << "Input playback startPosition ( 1 to 10 ) : ";
    cin >> startPosition;
    cout << "Input playback direction ( 1-Forward, 2-Backward ) : ";
    cin >> playDirection;

    file.open( "test.dat", ios::in | ios::binary);

    /* Hints
    long int i;
    file.seekg( i, ios::beg );

    sets read position at i-th byte from the beginning of file

    -----------------------
    int number;
    file.read( reinterpret_cast<char *>( &number ), sizeof( number ) );

    reads an integer from file.

    -----------------------
    long int j;
    file.seekg( -j, ios::cur );

    sets the read position j bytes before the current position
    */









    file.close();
    return 0;
}
```

End of Page